

Cellular automaton

Chris Theis

1.0 Introduction

A cellular automaton consists of a grid with N cells, that can represent r different states $\sigma_i = 0 \dots r-1$. The state of cell i at the time $t + 1$ is determined by k other cells at the time t according to a specified rule:

$$\sigma_i(t + 1) = f_i(\sigma_j(t), j=1 \dots k) \quad (\text{EQ 1})$$

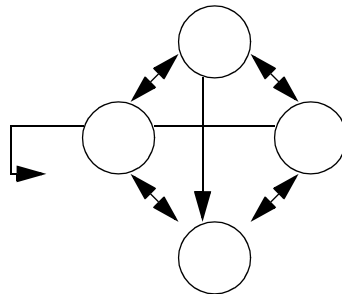


FIGURE 1. A cellular automaton with three input cells

Consequently a cellular automaton shows the following properties:

- discrete space, discrete time steps, discrete states
- cell state at $t+1$ determined by a rule with k input values
- all cells are processed in parallel

1.1 A one dimensional cellular automaton

A 1-D cellular automaton is represented by a chain of N cells and a rule that determines the cell state (0 or 1) at $t + 1$ using the input from the two neighbors.

$$\sigma_i(t + 1) = f_i(\sigma_{j-1}(t), \sigma_j(t), \sigma_{j+1}(t)) \quad (\text{EQ 2})$$

Looking at the possible input combinations with the corresponding results for three different rules, we obtain the following

TABLE 1. Rules for a cellular automaton

Inputs	111	110	101	100	011	010	001	000
Rule 8	0	0	0	0	1	0	0	0
Rule 22	0	0	0	1	0	1	1	0
Rule 90	0	1	0	1	1	0	1	0

The name of the rules originate from their decimal representation (00010110 binary = 22 decimal). In principle cellular automaton can be grouped with respect to their time development:

- After some time all 1's have disappeared
- After a long time the automaton shows a stationary structure, which means that the cell states don't change any more or have a short period.
- The cellular automaton shows chaotic behavior.
- Large structures appear.

Chaotic behavior can be observed if two configurations, that are originally close within the phasespace, drift appart exponentially as the time proceeds. It is necessary to define a distance within the phasespace to measure and describe chaotic behavior. The *Hamming* distance is defined by

$$d(t) = \frac{1}{N} \sum \sigma_i^A \otimes \sigma_i^B \quad (\text{EQ 3})$$

(\otimes denotes exclusive or).

As t goes towards infinity d should go towards 0 for a "frozen" system, otherwise the system shows chaotic behavior.

1.2 Implementation

Every binary rule can be written using logical operators

e.g. Rule 8 $f_i(\sigma_{j-1}(t), \sigma_j(t), \sigma_{j+1}(t)) = \overline{\sigma_{i-1}} \wedge \sigma_i \wedge \sigma_{i+1}$. The best way to implement the rules is to store them in an array f(n) using n as an index. The corresponding index to specified input values can be calculated with

$$n = shl(shl(\sigma_{i-1}) \wedge \sigma_i) \wedge \sigma_{i+1} \quad (\text{EQ 4})$$

(shl ... denotes a binary shift left operation)

The calculated index n is used to obtain the new cell state from the look up table.

1.3 A 2-D cellular automaton (Q2R)

The Q2R automaton is a 2 dimensional cellular automaton implemented by Vichniac 1984. There are two possible cell states {0,1} determined by the rule:

$$\sigma_{ij}(t+1) = f_{ij}(x_{ij}) \otimes \sigma_{ij}(t) \quad (\text{EQ 5})$$

$$x_{ij} = \sum_{nn} \sigma = \sigma_{i-1j} + \sigma_{i+1j} + \sigma_{ij-1} + \sigma_{ij+1} \quad (\text{EQ 6})$$

$$f(x) = \begin{cases} 1 & (x = 2) \\ 0 & (x \neq 2) \end{cases} \quad (\text{EQ 7})$$

Q2R is very useful for microcanonical simulations but you have to keep in mind that not the whole phasespace is covered but only the configurations of a cycle. Therefore using one run of the simulation for the Ising model may result in an equilibrium that is not the global minimum of the energy or magnetization. Therefore several runs have to be made to cover the whole phasespace in contrary to Monte Carlo simulations (assumed that enough iterations have been made!). The advantage is the generally shorter runtime.